Ruby - Bug #7107

Ruby can no longer find constants in methods in anonymous modules

10/05/2012 10:06 AM - drbrain (Eric Hodel)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:	2.0.0		
ruby -v:	ruby 2.0.0dev (2012-09-06 trunk 36915) [x86_64-darwin12.1.0]	**Backport:**	

Description

=begin
With ruby 1.9 and newer C cannot be found if m is duplicated:

module M
C = 1

```
def self.m
  C
end
```

end

puts 'named module'
M.m

puts 'anonymous module'
m = M.dup
m.m

Ruby 1.8:

$ ruby -v t.rb
ruby 1.8.7 (2012-02-08 patchlevel 358) [universal-darwin12.0]
named module
anonymous module

With Ruby 1.9:

$ ruby19 -v t.rb
ruby 1.9.3p194 (2012-04-20 revision 35410) [x86_64-darwin12.2.0]
named module
anonymous module
t.rb:5:in m': uninitialized constant Module::C (NameError) from t.rb:14:in '

With trunk:

```
$ ruby19 -v t.rb
ruby 1.9.3p194 (2012-04-20 revision 35410) [x86_64-darwin12.2.0]
named module
anonymous module
t.rb:5:in `m': uninitialized constant Module::C (NameError)
 from t.rb:14:in `<main>'
```

({{m::C}}) works in all three versions, though.
=end

Associated revisions

Revision a70bb8889b3130d72ec5a1f0ce92f386000a60da - 12/17/2012 09:49 AM - Charlie Somerville

- class.c (rewrite_cref_stack, clone_method): rewrite a method's cref

stack when cloning into a new class to allow lexical const lookup to
work as expected [ruby-core:47834] [Bug #7107]
- test/ruby/test_class.rb (class TestClass): related test

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@38423 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision a70bb888 - 12/17/2012 09:49 AM - Charlie Somerville

- class.c (rewrite_cref_stack, clone_method): rewrite a method's cref
stack when cloning into a new class to allow lexical const lookup to
work as expected [ruby-core:47834] [Bug #7107]
- test/ruby/test_class.rb (class TestClass): related test

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@38423 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 10/06/2012 05:47 AM - drbrain (Eric Hodel)

=begin
Additionally, when the module is given a name it still can't find the constant:

```
module M
C = 1

def self.m
  C
end

end

puts 'named module'
M.m

puts 'anonymous module'
m = M.dup
begin
m.m
rescue NameError
p $!
end

puts 're-named module'
N = m
begin
N.m
rescue NameError
p $!
end
```

Ruby trunk:

```
$ ruby20 -v t.rb
ruby 2.0.0dev (2012-09-06 trunk 36915) [x86_64-darwin12.1.0]
named module
anonymous module
#<NameError: uninitialized constant Module::C>
re-named module
#<NameError: uninitialized constant Module::C>
```

=end

#2 - 10/27/2012 09:31 AM - ko1 (Koichi Sasada)

- Assignee set to mame (Yusuke Endoh)

#3 - 10/27/2012 07:09 PM - mame (Yusuke Endoh)

- Status changed from Open to Assigned

- Assignee changed from mame (Yusuke Endoh) to nobu (Nobuyoshi Nakada)

Indeed, it looks a bug. Nobu, could you investigate?

--

Yusuke Endoh mame@tsg.ne.jp

#4 - 12/13/2012 05:54 AM - tarui (Masaya Tarui)

hi,

I found strange behavior.

```
$ ruby -e "module M;C=1;def self.f;C end end;d=M.dup;p M.f;p d.f;class A;end;p d.f"
1
1
-e:1:in f': uninitialized constant Module::C (NameError) from -e:1:in '
```

It seems to hold the problem in InlineCache too.

#5 - 12/13/2012 06:29 AM - tarui (Masaya Tarui)

additional sample.
d.f referring to M::C is correct? or d::C?

```
$ ruby -e "module M;C=1;def self.f;C end end;d=M.dup;p M.f;p d.f;d::C=2;p M.f;p d::C;p d.f"
1
1
-e:1: warning: already initialized constant #Module:0x007ffb39f4c860::C
-e:1: warning: previous definition of C was here
1
2
1
```

#6 - 12/15/2012 12:25 PM - Anonymous

=begin

Nobu, I have found the cause of the bug - the cref_stack of methods are not fixed up to point to the new class/module when the class/module is duped.

Do you want me to commit it or attach it here for your review?

=end

#7 - 12/17/2012 11:38 AM - Anonymous

- Assignee changed from nobu (Nobuyoshi Nakada) to Anonymous

#8 - 12/17/2012 06:49 PM - Anonymous

- Status changed from Assigned to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r38423.
Eric, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- class.c (rewrite_cref_stack, clone_method): rewrite a method's cref stack when cloning into a new class to allow lexical const lookup to work as expected [ruby-core:47834] [Bug #7107]
- test/ruby/test_class.rb (class TestClass): related test